

## CLAIMS

What is claimed is:

- 1 1. A method of maintaining version compatibility between a first computer program  
2 module and one or more interacting computer program modules that interact with the first  
3 module through an interface, wherein the modules are stored in computer storage, the method  
4 comprising the computer-implemented steps of:  
5 creating and storing first information describing the interface at a plurality of times  
6 with a corresponding plurality of instances of a data structure;  
7 creating and storing a mapping that associates the plurality of instances with a  
8 corresponding plurality of version numbers for the first module;  
9 automatically developing a second version number for a second module of the one or  
10 more interacting modules based on the plurality of instances of the data  
11 structure and the mapping;  
12 determining compatibility of the modules based on a first version number for the first  
13 module and the second version number for the second module.
- 1 2. A method as recited in Claim 1, wherein the step of automatically developing  
2 comprises:  
3 describing a subset of the interface, which subset is employed by the second module;  
4 determining of the plurality of instances at least one instance including data  
5 describing the subset of the interface; and  
6 assigning the second version number for the second module based on the mapping  
7 and the at least one instance.
- 1 3. A method as recited in Claim 2, wherein said step of assigning the second version  
2 number comprises assigning as the second version number a particular value of the plurality  
3 of version numbers for the first module, the particular value associated with a particular  
4 instance of the at least one instance, the particular instance corresponding to an earliest time  
5 of one or more times corresponding to the at least one instance.

1 4. A method as recited in Claim 1, wherein the plurality of version numbers for the first  
2 module corresponding to the plurality of instances vary in one direction with time of the  
3 plurality of times corresponding to the plurality of instances.

1 5. A method as recited in Claim 1, wherein said step of describing the interface includes  
2 generating and storing in a first instance of the data structure data indicating signatures of a  
3 plurality of routines of the interface at a first time, wherein a signature of each routine  
4 includes a name of the routine and a type of the routine and parameter types for all  
5 parameters of the routine.

1 6. A method as recited in Claim 1, wherein:  
2 each of the interacting computer program modules include instructions causing one or  
3 more processors to obtain at least one of a property of a corresponding  
4 networking device type of a plurality of networking devices types and an  
5 action performed by the corresponding networking device type;  
6 the first computer program module includes instructions causing one or more  
7 processors, based on interacting with a particular interacting computer  
8 program module, to perform at least one of communicating with a first device  
9 of the corresponding networking device type on a network of networking  
10 devices including the first device, and presenting properties of the first device  
11 to a network manager for the network, and displaying connections among the  
12 networking devices of the network to the network manager; and  
13 the plurality of networking devices types include one or more models of repeater, a  
14 switch, a router, a hub, a bridge, and a gateway.

1 7. A method as recited in Claim 1, wherein the second version number is developed  
2 when the second module is developed; and compatibility is determined at a later time.

- 1 8. A method of maintaining version compatibility between a first module and one or  
2 more interacting modules that interact with the first module through an interface, the method  
3 comprising the steps of:  
4 retrieving data from a stored mapping between a plurality of instances of a data  
5 structure describing the interface at a corresponding plurality of times and a  
6 corresponding plurality of version numbers for the first module;  
7 automatically developing a second version number for a second module of the one or  
8 more interacting modules based on the mapping;  
9 determining compatibility based on a first version number for the first module and the  
10 second version number for the second module.
- 1 9. A method as recited in Claim 8, said step of developing the second version number  
2 further comprising the steps of:  
3 describing a subset of the interface, which subset is employed by the second module;  
4 determining of the plurality of instances at least one instance including data  
5 describing the subset of the interface; and  
6 assigning the second version number for the second module based on the mapping  
7 and the at least one instance.
- 1 10. A method as recited in Claim 9, wherein said step of assigning the second version  
2 number comprises assigning as the second version number a particular value of the plurality  
3 of version numbers for the first module, the particular value associated with a particular  
4 instance of the at least one instance, the particular instance corresponding to an earliest time  
5 of one or more times corresponding to the at least one instance.
- 1 11. A method as recited in Claim 8, wherein the plurality of version numbers for the first  
2 module corresponding to the plurality of instances vary in one direction with time of the  
3 plurality of times corresponding to the plurality of instances.

1 12. A method as recited in Claim 8, wherein a first instance of the data structure  
2 comprises data indicating a routine name and a routine type of a routine of the first module at  
3 a first time.

1 13. A method as recited in Claim 12, wherein the first data further indicates a parameter  
2 type for the routine.

1 14. A method as recited in Claim 8, wherein a first instance of the data structure  
2 comprises data indicating signatures of a plurality of routines of the interface at a first time,  
3 wherein a signature of each routine includes a name of the routine and a type of the routine  
4 and parameter types for all parameters of the routine.

1 15. A method as recited in Claim 14, wherein the data indicating signatures of the  
2 plurality of routines of the interface at a first time comprises hashed values, each hashed  
3 value uniquely indicating a signature of each routine of the interface.

1 16. A method as recited in Claim 14, wherein the plurality of routines comprises all the  
2 routines of the interface.

1 17. A method as recited in Claim 14, wherein the plurality of routines comprises all the  
2 routines of the interface except routines not implemented in the first module.

1 18. A method as recited in Claim 8, wherein:  
2 the second version number is developed when the second module is developed; and  
3 compatibility is determined at a later time.

1 19. A method as recited in Claim 18, wherein compatibility is determined when the  
2 second module is installed for use with the first module.

1 20. A method as recited in Claim 18, wherein compatibility is determined when the  
2 second module is invoked for execution by the first module.

1 21. A method as recited in Claim 8, wherein each module of the first module and the one  
2 or more interacting modules comprises instructions for causing one or more processors to  
3 perform one or more tasks.

1 22. A method as recited in Claim 8, wherein the first module comprises instructions for  
2 causing one or more processors to manage a plurality of networking devices in response to  
3 data indicating input by a user.

1 23. A method as recited in Claim 8, wherein each interacting module of the one or more  
2 interacting modules comprises instructions for causing one or more processors to provide  
3 device-specific information for one of a plurality of networking devices.

1 24. A method as recited in Claim 21, wherein:  
2 each of the interacting modules include instructions causing one or more processors to  
3 obtain at least one of a property of a corresponding networking device type of  
4 a plurality of networking devices types and an action performed by the  
5 corresponding networking device type;  
6 the first module includes instructions causing one or more processors, based on  
7 interacting with a particular interacting computer program module, to perform  
8 at least one of communicating with a first device of the corresponding  
9 networking device type on a network of networking devices including the first  
10 device, and presenting properties of the first device to a network manager for  
11 the network, and displaying connections among the networking devices of the  
12 network to the network manager; and  
13 the plurality of networking devices types include one or more models of repeater, a  
14 switch, a router, a hub, a bridge, and a gateway.

- 1 25. A method of determining version compatibility between a first module and a second  
2 module of one or more interacting modules that interact with the first module through an  
3 interface, the method comprising the steps of:  
4 obtaining a first version number for the first module;  
5 obtaining a second version number for the second module, the second version number  
6 set when the second module is developed based on a mapping between a  
7 plurality of instances of a data structure describing the interface at a  
8 corresponding plurality of times and a corresponding plurality of version  
9 numbers for the first module; and  
10 determining whether the modules are compatible based on the first version number  
11 and the second version number.
- 1 26. A method as recited in Claim 25, wherein:  
2 each of the interacting modules include instructions causing one or more processors to  
3 obtain at least one of a property of a corresponding networking device type of  
4 a plurality of networking devices types and an action performed by the  
5 corresponding networking device type;  
6 the first module includes instructions causing one or more processors, based on  
7 interacting with a particular interacting computer program module, to perform  
8 at least one of communicating with a first device of the corresponding  
9 networking device type on a network of networking devices including the first  
10 device, and presenting properties of the first device to a network manager for  
11 the network, and displaying connections among the networking devices of the  
12 network to the network manager; and  
13 the plurality of networking devices types include one or more models of repeater, a  
14 switch, a router, a hub, a bridge, and a gateway.

- 1 27. A computer-readable medium for maintaining version compatibility between a first  
2 module and one or more interacting modules that interact with the first module through an  
3 interface, the computer-readable medium carrying:  
4 a plurality of instances of a data structure describing the interface at a corresponding  
5 plurality of times;  
6 a mapping that associates the plurality of instances with a corresponding plurality of  
7 version numbers for the first module; and  
8 one or more sequences of instructions, which, when executed by one or more  
9 processors, cause the one or more processors to carry out the steps of  
10 retrieving data from the mapping, and  
11 developing a second version number for a second module of the one or more  
12 interacting modules,  
13 wherein compatibility is determined based on a first version number for the first  
14 module and the second version number for the second module.
- 1 28. A system for maintaining version compatibility between a first module and one or  
2 more interacting modules that interact with the first module through an interface, the system  
3 comprising:  
4 means for retrieving a stored mapping between a plurality of instances of a data  
5 structure describing the interface at a corresponding plurality of times and a  
6 corresponding plurality of version numbers for the first module;  
7 means for automatically developing a second version number for a second module of  
8 the one or more interacting modules based on the mapping; and  
9 means for determining compatibility based on a first version number for the first  
10 module and the second version number for the second module.
- 1 29. A computer system for maintaining version compatibility between a first module and  
2 one or more interacting modules that interact with the first module through an interface, the  
3 system comprising:  
4 a processor;

5 a computer-readable medium carrying  
6 a stored mapping between a plurality of instances of a data structure  
7 describing the interface at a corresponding plurality of times and a  
8 corresponding plurality of version numbers for the first module, and  
9 one or more stored sequences of instructions which, when executed by the  
10 processor, cause the processor to carry out the steps of:  
11 retrieving data from the mapping;  
12 developing a second version number for a second module of the one or  
13 more interacting modules based on the mapping; and  
14 determining compatibility based on a first version number for the first  
15 module and the second version number for the second module.